

Uniwersytet Jagielloński

Jacek Lembas, Rafał Kawa

Wstęp do informatyki

Kraków 2011/2012

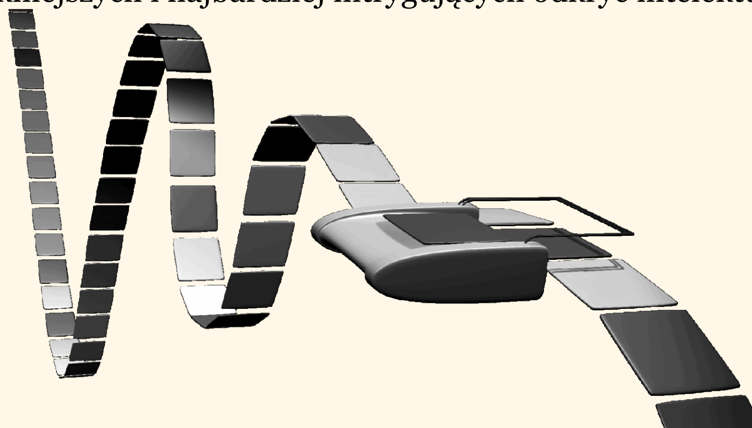
Spis treści

1 Modele obliczeń

1.1 Maszyna Turinga

1.1.1 Wprowadzenie

W roku 1936 wspominany wcześniej Alan Turing opublikował jeden z efektów swoich prac dotyczących obliczalności - czyli mówiąc potocznie tego, co jest możliwe do obliczenia automatycznie. Zajmował się wtedy między innymi opracowaniem różnorodnych teoretycznych modeli maszyn, z których jeden nazywany jest dziś **maszyną Turinga**, a co uznawane jest za jedno z najpiękniejszych i najbardziej intrygujących odkryć intelektualnych XX wieku[1].



Rysunek 1.1 Szeroko znana artystyczna wizja maszyny Turinga

W historycznie pierwszej opublikowanej wersji maszyna Turinga jest koncepcyjnym (dziś pewnie użyto by słowa wirtualnym) urządzeniem składającym się po pierwsze z obustronnie nieskończonej taśmy podzielonej na klatki z zapisanymi w nich pojedynczymi znakami. Po drugie z operującej na taśmie głowicy mającej w jednym cyklu (kroku) pracy możliwość: odczytania znaku z aktualnej widocznej klatki taśmy, zapisania (nadpisania) znaku w aktualnie widoczną klatkę taśmy, przesunięcia się o jedną klatkę w prawo lub w lewo i ewentualnie pozostania głowicy nieruchomą. Ponadto, oprócz tak określonego *hardwer*-u maszyna Turinga posiada *software* w postaci zestawu reguł określających zachowanie maszyny (w rozumieniu zapisywanego znaku oraz charakteru ruchu głowicy) w zależności od odczytanego znaku.

[1] *The Turing machine is one of the most beautiful and intriguing intellectual discoveries of the 20th century.*

Możliwe są różnorodne modyfikacje maszyny Turinga - od czysto technicznych (w rodzaju zastąpienia poruszającej się po taśmie głowicy wersją z nieruchomą głowicą, ale ruchomą taśmą) po zmiany pozornie istotniejsze. Na przykład taśma może być ograniczona z jednej strony, ale wtedy klatki mogą naprzemiennie odpowiadać lewej i prawej stronie ogranicznika czyniąc modyfikację równoważną. Liczbę dostępnych symboli możemy ograniczyć do jedynie dwu - wtedy binarna reprezentacji każdego wyjściowego symbolu również czyni modyfikację równoważną. Ponadto, oprócz modyfikacji pozornie ograniczających możliwe są zmiany pozornie rozszerzające - na przykład do wersji z wieloma taśmami, co również może być zastąpione równoważną maszyną jednotaśmową. Można ograniczyć determinizm przez dopuszczenie wielu możliwych zachowań przy odczycie jednego symbolu, ale i wtedy nie otrzymujemy maszyny o zwiększonej mocy obliczeniowej. Nawet zwiększenie wymiaru taśmy (w przypadku prostokąta otrzymamy model zwany **mrówką Langtona**) nie zwiększa możliwości w stosunku do maszyny Turinga z jedną dwustronnie nieograniczoną taśmą z jedną głowicą.

Jednakże istota maszyny Turinga tkwi w czymś innym. Otóż okazuje się, że maszyna Turinga jest modelem obliczeniowym, a więc pewnym teoretycznym komputerem i co więcej - wszystkie realnie działające komputery mogą być zastąpione odpowiednią maszyną Turinga. Co ciekawe, mimo swojej prostoty moc obliczeniowa maszyny Turinga jest nawet większa niż każdego aktualnie zrealizowanego komputera.

1.1.2 Formalna definicja

Formalnie maszyna Turinga definiowana jest przez pewną krotkę wraz z dodatkowymi warunkami. Liczność krotki zależy od upodobań autorów i podczas gdy jedni ograniczają liczbę elementów dokładając warunki, drudzy włączają warunki jako kolejne elementy w krotce. Ponadto, nawet przy identyczność elementów w krotce występują różnice w kolejności. Dlatego w naszej wersji po prostu wymienimy składowe formalnej definicji, opiszemy warunki, wprowadzimy przydatne oznaczenia - poprzestając na stronie merytorycznej. I tak na maszynę Turinga składa się:

- Niepusty, skończony, uporządkowany zbiór stanów – oznaczany przez Q. Maszyna Turinga zawsze znajduje się w którymś ze stanów i w pojedynczym cyklu pracy może nastąpić zmiana stanu na inny. Ponadto, z racji stosowania w zapisach poszczególnych stanów pojedynczych znaków oraz przy założeniu że zbiór stanów musi być skończony i uporządkowany mówimy o **alfabecie stanów**.
- Spośród wszystkich stanów wyróżniamy jeden szczególny stan (oznaczany przez q_0) zwany **stanem początkowym** i jest to stan w którym znajduje się maszyna Turinga na początku pracy.
- Ze zbioru stanów wyróżniony jest podzbiór **stanów końcowych** (oznaczany przez F), czyli stanów powodujących zatrzymanie się maszyny Turinga. Niektórzy autorzy przyjmują, że stanów końcowych może być tylko jeden, co nie stanowi istotnego ograniczenia. Mając wiele stanów końcowych możemy przyjąć, że nie są one końcowe, ale

jedynym możliwym działaniem z nimi związanym będzie przejście do jednego specjalnie stworzonego stanu ostatecznie końcowego.

- Maszyna Turinga bywa wykorzystywana nie tylko do obliczeń, ale również do automatycznego uzyskania odpowiedzi na różnorodne pytania. Dlatego niektórzy autorzy dzielą zbiór stanów końcowych na podzbiór stanów o pozytywnym zakończeniu i podzbiór stanów końcowych o negatywnym zakończeniu. Podobnie, chcąc by maszyna Turinga reagowała świadomie na sytuacje niecodzienne (błąd danych, nieprzewidziany dla danego stanu symbol) wydziela się ze stanów końcowych specjalne stany świadczącego o wyjątkowości. Tym samym odróżniamy przewidywane poprawne zakończenie od działania przerwanego nieprzewidzianymi przyczynami.
- Zbiór możliwych symboli taśmy oznaczany będzie przez Γ i musi być niepusty, skończony i uporządkowany. Tym samym usprawiedliwione będzie nazwanie go **alfabetem taśmy**.
- Spośród możliwych symboli taśmy wyróżniamy oznaczany przez b symbol pusty (*blank*) . Możemy też przyjąć, że taśma bez żadnych ingerencji zapełniona jest właśnie symbolem pustym. Występuje pewna dowolność w wyborze oznaczenia symbolu pustego i prawie nigdy nie jest to znak spacji – jako niemożliwy do odróżnienia od zwykłego odstępu. W literaturze przedmiotu dość często stosowany jest znak #, bywa też stosowany znak ., różne greckie litery – my zaś będziemy się posługiwać najczęściej znakiem _.
- Ze względów praktycznych stosowane jest specjalne oznaczenie Σ dla zbioru symboli taśmy bez symbolu pustego, co nazywane jest **alfabetem wejściowym**.
- Na oznaczenie możliwych ruchów głowicy stosuje się najczęściej pojedyncze znaki pochodzące od słów opisujących charakter ruchu w używanym języku naturalnym. Stąd litery L(*left*) i R(*right*) dla oznaczenia ruchów w lewo i w prawo oraz litera S(*the same*) lub N(*none*) dla braku ruchu głowicy, zastępowane znakiem spacji lub poziomej kreski. Pisząc po polsku byłyby to litery odpowiednio L, P oraz N, natomiast w naszym przypadku zastosujemy znaki strzałek odpowiednio ← i → dla ruchów oraz znaku ↓ dla sytuacji pozostawania głowicy na miejscu.
- Działanie maszyny Turinga (w rozumieniu jej sterowania) opisane jest zestawem (skończonym zbiorem) reguł (oznaczanym przez δ), co do zapisu których występuje pewna dowolność (mimo merytorycznej równoważności). Dość często pojedyncza reguła przedstawiana jest jako piątka złożoną z zapisu aktualnego stanu, aktualnie czytanego symbolu (znaku nad którym jest głowica), nadpisywanego symbolu, zapisu ruchu głowicy i zapisu nowego stanu. Merytorycznie występuje wyraźny rozdział na dwa pierwsze elementy i trzy ostatnie, zaś a w ramach poszczególnych części stosowana jest różna kolejność. W naszym przypadku przyjmujemy:

$$\delta \in (Q \times \Gamma) \times \left(\Gamma \times \{ \leftarrow, \rightarrow, \downarrow \} \times Q \right)$$

i zbiory użyte w szczególnym iloczynie kartezjańskim opisane są powyżej zgodnie z kolejnością. Dwa pierwsze

zgrupowane zbiory odzwierciedlają bieżącą konfigurację maszyny, podczas gdy trzy następne opisują nową konfigurację wraz z koniecznym działaniem. Stąd na sterowanie maszyny Turinga możemy spojrzeć jak na czysto matematycznie traktowaną dwuelementową relację bieżącej konfiguracji maszyny oraz konfiguracji po wykonaniu ruchu głowicy.

Zauważmy ponadto, że powyższe określenie sterowania dopuszcza sytuację przyporządkowania tej samej bieżącej konfiguracji dwu różnych konfiguracji po wykonaniu ruchu głowicy. Tym samym, dopuszczalny jest niedeterminizm w działaniu maszyny nazywanej z tego powodu **niedeterminstyczną maszyną Turinga**.

- Wymóg determinizmu oznacza w naszej sytuacji związanie z każdą bieżącą konfiguracją dokładnie jednej sytuacji po ruchu głowicy co oznacza, że relacja δ będzie po prostu funkcją. Ponieważ maszyny Turinga z założonym determinizmem mają znacznie większe zastosowanie, dlatego znacznie powszechniejszy jest „funkcyjny" zapis sterowania w postaci:

$$\delta : Q \times \Gamma \to \Gamma \times \{\leftarrow, \rightarrow, \downarrow\} \times Q$$

  Różnice w sposobie prezentacji powyższej funkcji (czy też funkcji jako takiej) są przyczyną pozornie różnych zapisów, zaś maszyna Turinga ze sterowanie będącym funkcją nazywana jest **deterministyczną maszyną Turinga**.
- Niezależnie od determinizmu, do zapisu sterowania wykorzystuje się postać stablicowaną i mówimy wtedy o **tablicy sterującej** maszyny Turinga.
- Ujednoznacznienie wymaga jeszcze położenie głowicy względem danych (symboli) wejściowych na wstępie działań. Przyjmujemy zatem, że w stanie początkowym głowica znajduje się na pierwszej od lewej klatce z niepustym symbolem.

By pozostać w zgodzie ze sposobami definiowania maszyny Turinga spotykanymi w literaturze i internecie podsumujmy. Maszyna Turinga obejmuje:

Q - alfabet stanów

$q_0 \in Q$ - wyróżniony stan początkowy

$F \subset Q$ - podzbiór stanów końcowych

Γ - alfabet symboli taśmy

$b \in \Gamma$ - wyróżniony (pusty) symbol taśmy

$\Sigma = \Gamma \setminus \{b\}$ - podalfabet symboli wejściowych

$\delta : Q \times \Gamma \to \Gamma \times \{\leftarrow, \rightarrow, \downarrow\} \times Q$ - funkcję (relację) sterująca

1.1.3 Przykłady

Przykład 1. Pracowity bóbr stopnia trzeciego

Określmy maszynę Turinga o zbiorze stanów $Q = \{ \mathsf{A}, \mathsf{B}, \mathsf{C}, \mathsf{Z} \}$, stanie początkowym $q_0 = \mathsf{A}$, stanami końcowymi $F = \{ \mathsf{Z} \}$, operującą na taśmie z dwoma możliwymi symbolami $\Gamma = \{ 0, 1 \}$, z symbolem pustym $b = 0$ oraz funkcją sterującą:

$$\delta = \begin{Bmatrix} (\mathsf{A}, 0) \mapsto (1, \rightarrow, \mathsf{B}), & (\mathsf{B}, 0) \mapsto (1, \leftarrow, \mathsf{A}), & (\mathsf{C}, 0) \mapsto (1, \leftarrow, \mathsf{B}), \\ (\mathsf{A}, 1) \mapsto (1, \leftarrow, \mathsf{C}), & (\mathsf{B}, 1) \mapsto (1, \rightarrow, \mathsf{B}), & (\mathsf{C}, 1) \mapsto (1, \rightarrow, \mathsf{Z}) \end{Bmatrix}$$

Zaprezentowana maszyna Turinga robi pozornie niewiele - jak przekonamy się dalej w szczególny sposób wypisuje sześć znaków 1. Jednakże jest przykładem klasy maszyn będących rozwiązaniem zadania zwanego problemem pracowitego bobra, postawionego następująco: przy danej liczbie stanów, dwuznakowym alfabecie taśmy, ruchach głowicy w prawo i w lewo określ maszynę Turinga działającą jak najdłużej ale nie w nieskończoność. Okazuje się, że w ogólnym przypadku zadanie jest niezwykle trudne, funkcja liczby kroków w zależności od ilości stanów nie jest obliczalna i rośnie szybciej niż większość znanych funkcji możliwych do wyliczenia. Tym niemniej, mimo niezwykle interesujących własności maszyna pracowitego bobra posłuży nam jedynie do demonstracji zagadnień związanych ze sposobami prezentacji działań maszyny Turinga jako takiej.

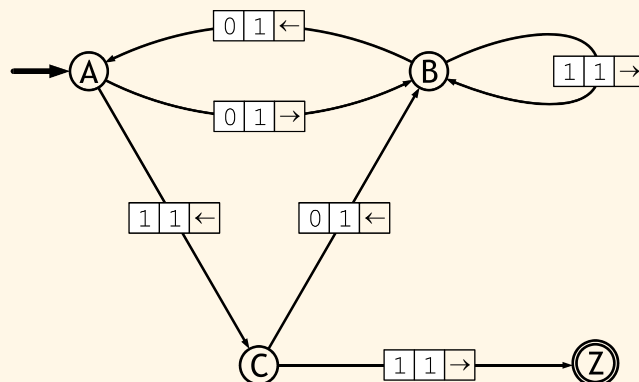
Przy ogólnym definiowaniu maszyny wspomnieliśmy o zapisie sterowania maszyny Turinga poprzez tak zwaną tablicę sterującą. Pola tablicy (tabeli) sterującej indeksowane są wszystkimi możliwymi stanami w jednym wymiarze oraz wszystkimi możliwymi symbolami taśmy w drugim wymiarze - zatem odpowiadają wszystkim możliwym parom stanu i czytanego symbolu, stanowiąc dziedzinę funkcji δ. Zarazem pola tabeli zawierają trójki mówiące o działaniu maszyny reprezentując wartości funkcji δ. Tego rodzaju zapis sterowania uwzględnia komplet możliwych sytuacji wejściowych, zatem wypełnienie całej tabeli gwarantuje pełną obsługę wszystkich możliwych sytuacji.

	A	B	C	Z
0	$(1, \rightarrow, \mathsf{B})$	$(1, \leftarrow, \mathsf{A})$	$(1, \leftarrow, \mathsf{B})$	
1	$(1, \leftarrow, \mathsf{C})$	$(1, \rightarrow, \mathsf{B})$	$(1, \rightarrow, \mathsf{Z})$	

Powyższa tabela jest właśnie tablicą sterującą dla określonej na wstępie maszyny Turinga pracowitego bobra.

Kolejnym sposobem prezentacji sterowania maszyny Turinga jest tak zwany **diagram stanów**, zaczerpnięty z działu informatyki zwanego teorią automatów. Diagram stanów maszyny Turinga to graf o wierzchołkach odpowiadających stanom (zwyczajowo rysowanych jako okręgi z symbolami stanów w środku) i krawędziami odpowiadającymi regułom sterowania. Jeżeli w regule sterowania występuje przejście od jednego stanu do innego, to w diagramie

stanów występuje krawędź łącząca oba stany skierowana od stanu aktualnego do stanu na końcu cyklu pracy. Ponieważ w funkcji sterującej przejście pomiędzy stanami jest zależne od aktualnego symbolu taśmy, może powodować wydrukowanie nowego symbol i określa ruch głowicy – dlatego krawędzie grafu etykietowane są opisaną trójką elementów. Ponadto stosuje się powszechnie przyjętą konwencje rysunkowego wyróżniania stanu początkowego przez skierowaną do niego strzałkę oraz stanów końcowych przez podwójny obrys kółka.



Rysunek 1.2 Diagram stanów maszyny Turinga pracowitego bobra stopnia trzeciego

Powyższy rysunek jest diagramem stanów maszyny Turinga trzystanowego pracowitego bobra, z widoczną odpowiedniością między parami wierzchołków i etykiet łączących je krawędzi a piątkami funkcji sterującej.

Do opisu działania maszyny Turinga dla konkretnych danych wykorzystuje się często specjalny diagram postępu obliczeń, który przy podanej zawartości taśmy z wyróżnioną pozycją głowicy i zapisanym aktualnym stanem w kolejnych wierszach podaje zmiany stanu, zawartości taśmy i zmiany położenia głowicy.

1.	A	0	0	0	0	**0**	0	0	0
2.	B	0	0	0	0	1	**0**	0	0
3.	A	0	0	0	0	**1**	1	0	0
4.	C	0	0	0	**0**	1	1	0	0
5.	B	0	0	**0**	1	1	1	0	0
6.	A	0	**0**	1	1	1	1	0	0
7.	B	0	1	**1**	1	1	1	0	0
8.	B	0	0	0	**1**	1	1	0	0

8.	B	0	1	1	**1**	1	1	0	0
9.	B	0	1	1	1	**1**	1	0	0
10.	B	0	1	1	1	1	**1**	0	0
11.	B	0	1	1	1	1	1	**0**	0
12.	A	0	1	1	1	1	**1**	1	0
13.	C	0	1	1	1	**1**	1	1	0
	Z	0	1	1	1	1	**1**	1	0

Powyższy diagram jest przykładem diagramu postępu wyjściowej maszyny Turinga dla problemu pracowitego bobra. Możemy krok po kroku prześledzić działanie zadanej maszyny wspomagając się diagramem stanów rozpoczynając od sytuacji początkowej. Na początku maszyna jest w stanie startowym A, odczytuje symbol 0, zatem sterowanie wydrukuje 1, przesunie głowicę w prawo i spowoduje przejście maszyny w stan B. Następnie maszyna będąc w stanie B i odczytując symbol 0 zgodnie ze sterowaniem wydrukuje symbol 1, przesunie głowicę w lewo i przejdzie w stan A. Tym razem maszyna będąc w stanie A odczytuje symbol 1, zatem po wydrukowaniu znaku 1 i ponownym ruchu głowicy w lewo znajdzie się w stanie C. Kontynuując, otrzymamy w stanie końcowym Z spójny ciąg sześciu symboli 1.

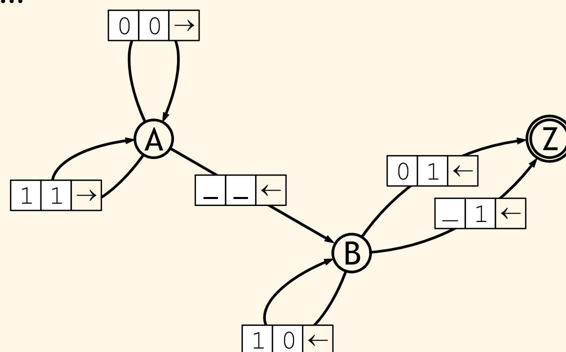
Przykład 2. Inkrementacja liczby binarnej
Spróbujmy określić maszynę Turinga, która zadaną na taśmie liczbę binarną zwiększy o jeden. Dopuszczalny zbiór symboli taśmy powinien obejmować co najmniej zbiór $\{_, 0, 1\}$ z symbolem pustym _ , zaś sama maszyna zapewne wymagała będzie co najmniej stanu startowego (oznaczmy go przez A) i symbolu końcowego (oznaczmy go przez Z). Zadanie możemy zrealizować w ten sposób, że algorytm (pamiętając iż na starcie głowica znajduje się na pierwszym niepustym symbolu) przejdzie głowicą w prawo omijając wszystkie symbole reprezentacji danej liczby by następnie wrócić i idąc w lewo od prawej strony odpowiednio nadpisywać klatki. Realizację omijania symboli danej liczby zrealizują reguły $(A, 0) \mapsto (0, \rightarrow, A)$ oraz $(A, 1) \mapsto (1, \rightarrow, A)$ powodujące przesunięcie w prawo bez względu na wczytany niepusty symbol. Powstaje zatem pytanie, co zrobić gdy maszyna będąc w stanie A odczyta symbol pusty? Konieczny wydaje się nowy stan (oznaczymy go przez B) do którego przejdziemy widząc w stanie A symbol _. Zarazem wiemy, że sytuacja odczytania znaku _ zdarzy się po odczytaniu całego ciągu znaków 0 lub 1, zatem napotkany znak _ będzie pierwszym symbolem pustym prawej stronie. Dlatego należy przesunąć głowicę w lewo, co spowoduje jej ustawienie na pierwszej cyfrze od prawej strony, zaś realizację zapewni reguła $(A, _) \mapsto (_, \leftarrow, B)$. Tym samym stan B będzie służył faktycznemu zwiększaniu danej liczby i okazuje się, że żaden więcej dodatkowy stan nie jest konieczny. Zastanówmy się, jak powinna reagować poszukiwana maszyna będąc w stanie B i odczytując symbol 0. Powinna wydrukować 1 i może zakończyć pracę, co zapewni reguła $(B, 0) \mapsto (1, \rightarrow, Z)$. Ale co będzie, kiedy maszyna w stanie B odczyta symbol 1? Powinna rzecz jasna wydrukować 0, ale merytorycznie wystąpiło przeniesienie zwiększające cyfrę na lewo od aktualnej, co wymaga odpowiedniej obsługi. Zauważmy jednak, że jeżeli po wydrukowaniu znaku 0 przesuniemy głowicę w lewo i napotkamy znak 0 to zastosowanie ma ostatnio określona reguła, natomiast jeżeli napotkamy znak 1 zastosowanie ma reguła właśnie omawiana. Tym samym reguła $(B, 1) \mapsto (0, \leftarrow, B)$ obsłuży wszystkie rozpatrywane sytuacje i pozostaje jeszcze odpowiedzieć na pytanie jak skończyć działanie, gdy na przykład powiększamy liczbę opisaną samymi znakami 1. Idąc od prawej strony poruszamy się w lewo nadrukowując znaki 0 i pozostając ciągle w sta-

nie B. Jednak kiedy symbole 1 się skończą napotkamy w stanie B znak pusty i zauważmy, że nadrukowanie znaku 1 zapewnia merytoryczną poprawność a zarazem jest sygnałem do zakończenia działania. Dlatego reguła $(\mathsf{B}, _) \mapsto (1, \leftarrow, \mathsf{Z})$ kompletuje pełne sterowanie poszukiwanej maszyny Turinga.

Podsumowując rozważania możemy opisać poszukiwaną maszynę Turinga stanami $Q = \{\mathsf{A}, \mathsf{B}, \mathsf{Z}\}$, ze stanem początkowym $q_0 = \mathsf{A}$, stanami końcowymi $F = \{\mathsf{Z}\}$, zbiorem symboli taśmy $\Gamma = \{_, 0, 1\}$, symbolem pustym $b = _$ i tablicą sterującą:

	A	B	Z
0	(0, →, A)	(1, ←, Z)	
1	(1, →, A)	(0, ←, B)	
_	(_, ←, B)	(1, ←, Z)	

Łatwo też wyrysować diagram stanów ...



Rysunek 1.3 Diagram stanów maszyny Turinga inkrementującej liczbę binarną

... jak na powyższym rysunku.
W uzupełnieniu skonfrontujmy powyższe rozważania z kilkoma diagramami postępu. I tak …

1.	A	_	**1**	0	1	0	0	_	_
2.	A	_	1	**0**	1	0	0	_	_
3.	A	_	1	0	**1**	0	0	_	_
4.	A	_	1	0	1	**0**	0	_	_
5.	A	_	1	0	1	0	**0**	_	_

5.	A	_	1	0	1	0	**0**	_	_
6.	A	_	1	0	1	0	0	**_**	_
7.	B	_	1	0	1	0	**0**	_	_
	Z	_	1	0	1	0	**1**	_	_

powyższy diagram obrazuje działanie dla danych postaci 10100 skutkujące przesuwaniem głowicy w prawo na koniec danej liczby z użyciem stanów A, zakończone w kroku 6. ruchem w lewo z przejściem w stan B zgodnie z regułą $(\mathsf{A}, _) \mapsto (_, \leftarrow, \mathsf{B})$. Z kolei w kroku 7. zastosowanie ma reguła $(\mathsf{B}, 0) \mapsto (1, \rightarrow, \mathsf{Z})$ prowadząc do stanu końcowego z pozostawieniem na taśmie odpowiedzi w postaci ciągu 10101.

1.	A	_	**1**	0	1	1	_	_
2.	A	_	1	**0**	1	1	_	_
3.	A	_	1	0	**1**	1	_	_
4.	A	_	1	0	1	**1**	_	_
5.	A	_	1	0	1	1	**_**	_
6.	B	_	1	0	1	**1**	_	_
7.	B	_	1	0	**1**	0	_	_
8.	B	_	1	**0**	0	0	_	_
	Z	_	1	**1**	0	0	_	_

Podobnie na powyższym diagramie kroki od 1. do 5. odpowiadają za przesunięcie głowicy w prawo na koniec danego ciągu symboli, jednakże w kroku 6. zastosowanie ma reguła $(\mathsf{B}, 1) \mapsto (0, \leftarrow, \mathsf{B})$ nie zaś reguła $(\mathsf{B}, 0) \mapsto (1, \rightarrow, \mathsf{Z})$ jak w poprzednim przykładzie. Reguła ma zastosowanie dwa razy z racji dwukrotnego odczytania przez głowicę symbolu 1 przy stanie B, natomiast reguła $(\mathsf{B}, 0) \mapsto (1, \rightarrow, \mathsf{Z})$ zastosowana w kroku 8. kończy działanie.

1.	A	_	_	**1**	1	1	_	_
2.	A	_	_	1	**1**	1	_	_
3.	A	_	_	1	1	**1**	_	_
4.	A	_	_	1	1	1	**_**	_
5.	B	_	_	1	1	**1**	_	_
6.	B	_	_	1	**1**	0	_	_
7.	B	_	_	**1**	0	0	_	_
8.	B	_	**_**	0	0	0	_	_
	Z	_	**1**	0	0	0	_	_

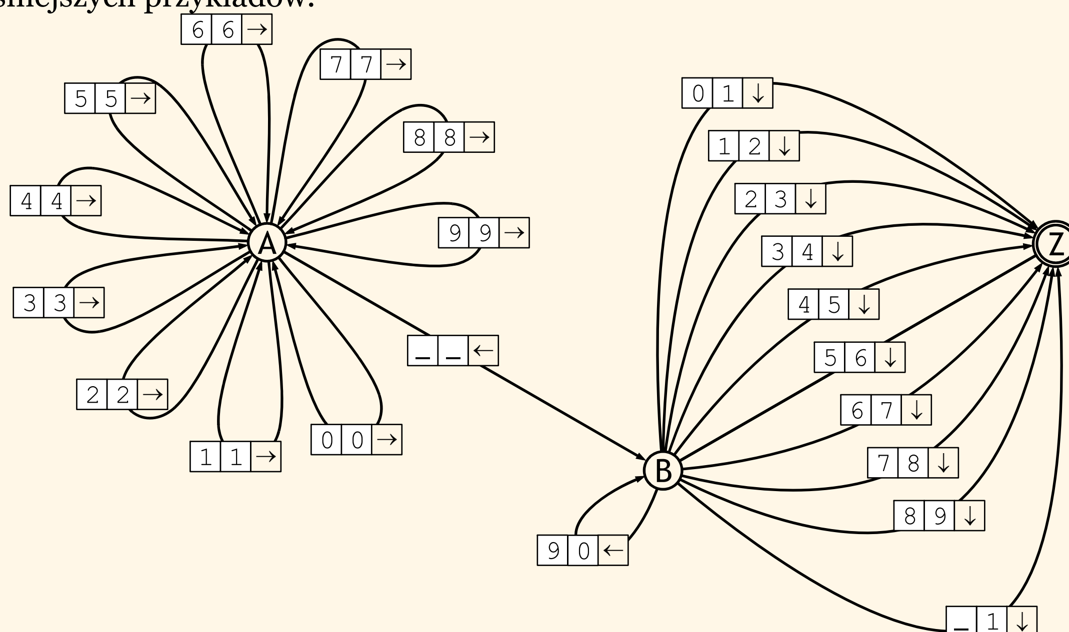
Ostatni diagram przedstawia zwiększenie liczby reprezentowanej przez ciąg symboli 1, z wielokrotnym zastosowaniem reguły $(\mathsf{B}, 1) \mapsto (0, \leftarrow, \mathsf{B})$ (aż do kroku 7.) oraz reguły $(\mathsf{B}, _) \mapsto (1, \leftarrow, \mathsf{Z})$ na zakończenie działania.

Przykład 3. Inkrementacja liczby dziesiętnej

Maszyna Turinga opisana w poprzednim przykładzie w swojej idei nadaje się do jednostkowego zwiększania liczb w zapisie o dowolnej podstawie - na przykład dziesiętnej. Stany nowej maszyny pozostaną bez zmian, to znaczy $Q = \{ \mathsf{A}, \mathsf{B}, \mathsf{Z} \}$, podobnie stan startowy $q_0 = \mathsf{A}$, stany końcowe $F = \{ \mathsf{Z} \}$. Natomiast alfabet taśmy rozszerzymy do dziesięciu cyfr otrzymując $\Gamma = \{ _, 0, 1, 2, 3, 4, 5, 6, 7, 8, 9 \}$ z symbolem pustym $b = _$.

	0	1	2	3	4	5	6	7	8	9	_
A	0 → A	1 → A	2 → A	3 → A	4 → A	5 → A	6 → A	7 → A	8 → A	9 → A	_ ← B
B	1 ↓ Z	2 ↓ Z	3 ↓ Z	4 ↓ Z	5 ↓ Z	6 ↓ Z	7 ↓ Z	8 ↓ Z	9 ↓ Z	0 ← B	1 ↓ Z
Z											

Tablicę sterującą przedstawia powyższa tabela – przy czym ze względów redakcyjnych wymiary są transponowane w stosunku do wcześniejszych przykładów.



Rysunek 1.4 Diagram stanów maszyny Turinga inkrementującej liczbę dziesiętną

Diagram stanów przedstawia powyższy rysunek, przy czym wielość symboli taśmy stawia rzecz całą na progu sensow-

ności. Jednakże, mimo wielości krawędzi widoczne jest ideowe a nawet graficzne podobieństwo do diagramu stanów z poprzedniego przykładu.

Tym samym podobnie jak w poprzednim przykładzie, stan A odpowiada za przesuwanie głowicy w prawo celem jej umiejscowienia za reprezentacją liczby, zaś stan B odpowiada za zmianę znaków na reprezentację po zwiększeniu. Startując ze stanu A i odczytując kolejne cyfry głowica przesuwa się w prawo, zaś kresem przesuwania jest odczyt symbolu pustego, po czym stosowana musi być reguła (A, _) ↦ (_, ←, B). Następnie ma miejsce zwiększanie cyfry o jeden i jeżeli nie wystąpi konieczność przeniesienia maszyna kończy pracę, zaś w przeciwnym wypadku przypadku zastosowanie znajduje reguła (B, 9) ↦ (0, ←, B) przesuwająca głowicę w lewo celem poszukiwania kolejnej cyfry do zwiększenia bez przeniesienia. W przypadku wyczerpania się cyfr zapisu przeniesienie dołoży jedynkę na początek, co zapewni reguła (B, _) ↦ (1, ←, Z).

1.	A	_	_	1	9	7	5	_	_
2.	A	_	_	1	9	7	5	_	_
3.	A	_	_	1	9	7	5	_	_
4.	A	_	_	1	9	7	5	_	_
5.	A	_	_	1	9	7	5	_	_
6.	B	_	_	1	9	7	5	_	_
	Z	_	_	1	9	7	6	_	_

1.	A	_	_	1	9	9	_	_
2.	A	_	_	1	9	9	_	_
3.	A	_	_	1	9	9	_	_
4.	A	_	_	1	9	9	_	_
5.	B	_	_	1	9	9	_	_
6.	B	_	_	1	9	0	_	_
7.	B	_	_	1	0	0	_	_
	Z	_	_	2	9	9	_	_

1.	A	_	_	_	9	9	_	_
2.	A	_	_	_	9	9	_	_
3.	A	_	_	_	9	9	_	_
4.	B	_	_	_	9	9	_	_
5.	B	_	_	_	9	0	_	_
6.	B	_	_	_	0	0	_	_
	Z	_	_	1	0	0	_	_

Diagramy postępu są podobne, wszystkie wymagają przejścia na prawą stronę reprezentacji z użyciem stanu A z cofnięciem się w lewo w sytuacji odczytu symbolu pustego. Samo wydrukowanie nowych cyfr może przebiegać różnie i na przykład pierwszy z powyższych diagramów obrazuje działanie polegające jedynie na zwiększeniu ostatniej cyfry. Z kolei środkowy diagram przedstawia sytuacje powodujące przeniesienia wymagające przesuwania głowicy w lewo dla poszukiwania cyfry nie powodującej przeniesienia. Ostatni, prawy diagram obrazuje działanie w sytuacji przeniesień wymagające dołożenia nowej cyfry przed wyjściową reprezentacją.

Przykład 4. Dekrementacja liczby binarnej

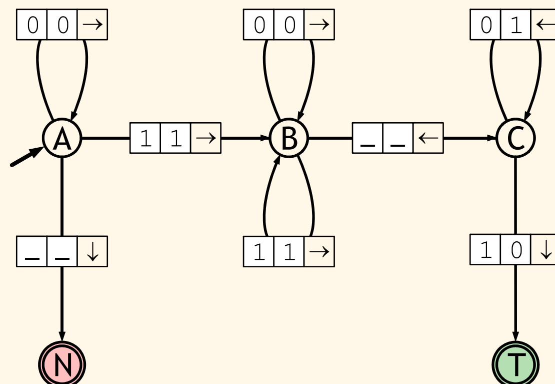
Jako kolejną przykładową maszynę Turinga rozpatrzmy maszynę odejmującą od zadanej dodatniej liczby binarnej wartość jednostkową. Idea algorytmu jest podobna - należy znaleźć się z głowicą na ostatniej cyfrze reprezentacji i gdy jest nią cyfra 1 wystarczy wydrukować 0, podczas gdy ostatnią cyfrą jest 0 należy w miejsce kolejnych wczytywanych znaków 0 wpisywać cyfry 1, jednakże kończąc na pierwszej wczytanej cyfrze 1 – zastąpionej znakiem 0. Jest to znane

ze szkolnego pisemnego odejmowania pożyczanie od cyfry z wyższej pozycji, ale w wersji binarnej.

Tym razem wprost podamy gotową maszynę, by następnie przeanalizować jej działanie. Określmy zatem pięć stanów $Q = \{ \mathsf{A}, \mathsf{B}, \mathsf{C}, \mathsf{T}, \mathsf{N} \}$, ze stanem początkowym $q_0 = \mathsf{A}$, ale z dwoma stanami końcowymi $F = \{ \mathsf{T}, \mathsf{N} \}$. Stan T będzie stanem poprawnego zakończenia odejmowania, natomiast stan N będzie obsługiwał sytuację niepoprawnych danych wejściowych. Alfabet taśmy określa zbiór $\Gamma = \{ _, 0, 1 \}$, symbol pusty $b = _$, zaś tablica sterująca ma postać:

	A	B	C	T	N
0	0 → A	0 → B	1 ← C		
1	1 → B	0 → B	0 ↓ N		
_	_ ↓ T	_ ← C			

Diagram stanów…



Rysunek 1.5 Diagram stanów maszyny Turinga dekrementującej liczbę binarną

… przedstawia powyższy rysunek.

Analizując diagram możemy zauważyć, że początkowe działanie maszyny w stanie A polega na ominięciu z przesuwaniem w prawo początkowych znaków 0 aż do wystąpienia znaku pustego lub znaku 1. Pierwszy przypadek oznacza zerowe dane, niemożliwe do odjęcia skąd przejście do końcowego stanu N sygnalizującego błąd.

1.	A	_	_	**0**	0	0	0	0	_	_
2.	A	_	_	0	**0**	0	0	0	_	_
3.	A	_	_	0	0	**0**	0	0	_	_
4.	A	_	_	0	0	0	**0**	0	_	_
5.	A	_	_	0	0	0	0	**0**	_	_
6.	A	_	_	0	0	0	0	0	**_**	_
	N	_	_	0	0	0	0	0	**_**	_

Przykładem opisanego działania jest powyższy diagram postępu.

Jeżeli jednak po ominięciu w stanie A początkowych znaków 0 napotkamy znak 1, to maszyna przejdzie do stanu B obsługującego sytuację poprawnych danych i zarazem zasadnicze działanie odejmowania. Będąc w stanie B głowica jest przesuwana w prawo omijając wszystkie pozostałe symbole reprezentacji liczby, aż do wystąpienia znaku pustego skutkującego przejściem do stanu C i ustawieniem głowicy na ostatnim znaku reprezentacji. Stan C służy faktycznemu wydrukowaniu reprezentacji po odjęciu, co w praktyce oznacza opisane wcześniej wydrukowanie znaku 0 gdy ostatnią cyfrą jest znak 1 i poprawne zakończenie przejściem do stanu T.

1.	A	_	_	**1**	0	1	1	_	_
2.	B	_	_	1	**0**	1	0	_	_
3.	B	_	_	1	0	**1**	1	_	_
4.	B	_	_	1	0	1	**1**	_	_
5.	B	_	_	1	0	0	1	**_**	_
6.	C	_	_	1	0	1	**1**	_	_
	T	_	_	0	1	1	**0**	_	_

Powyższy diagramem postępu obrazuje przykładowe działanie.

Możliwe jest jednak, że ostatnim znakiem reprezentacji będzie 0, a wtedy maszyna będąc w stanie C wydrukuje znak 1, ale przesunie głowicę w lewo celem poszukiwania cyfry do pożyczenia. Jeżeli będzie to znak 1 maszyna wydrukuje 0 i przejdzie do stanu T (czyli zadziała jak przy znaku 1 na końcu), natomiast gdy będzie to znak 0 konieczne będzie kontynuowanie pożyczania do jeszcze jednego znaku w lewo. Ponieważ wiemy, że co najmniej jeden znak 1 wy-

stępuje, działanie na pewno kiedyś się zakończy.

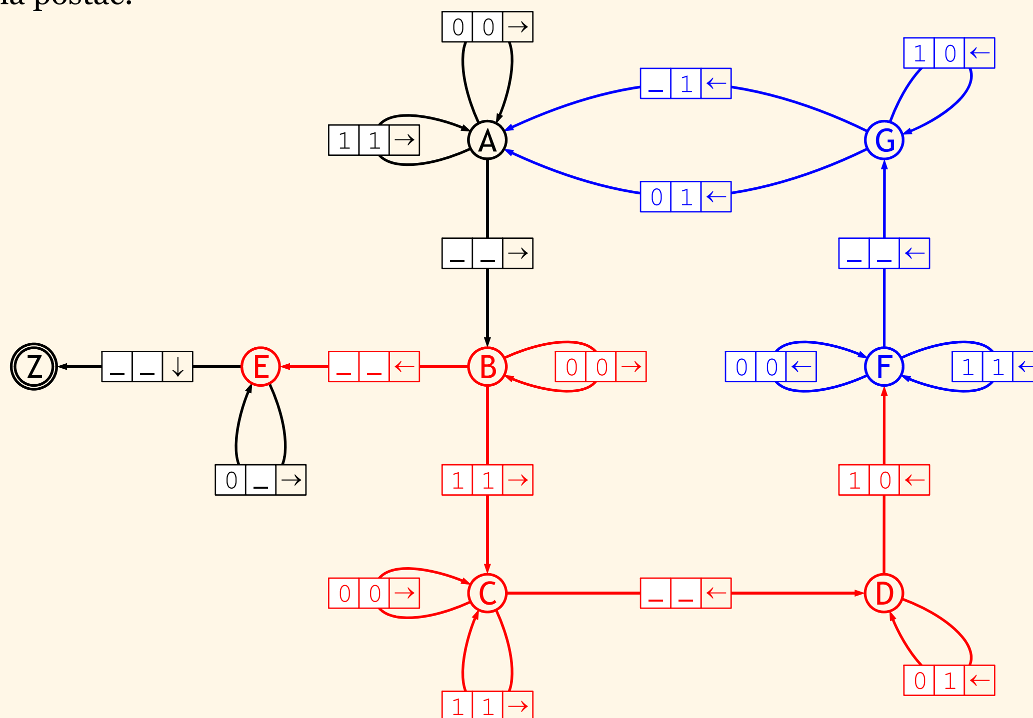
1.	A	_	_	**1**	0	1	0	0	_	_
2.	B	_	_	1	**0**	1	0	0	_	_
3.	B	_	_	1	0	**1**	0	0	_	_
4.	B	_	_	1	0	1	**0**	0	_	_
5.	B	_	_	1	0	1	0	**0**	_	_
6.	B	_	_	1	0	1	0	0	**_**	_
7.	C	_	_	1	0	1	0	**0**	_	_
8.	C	_	_	1	0	1	**0**	1	_	_
9.	C	_	_	1	0	**1**	1	1	_	_
	T	_	_	1	0	**0**	1	1	_	_

Przykładowe działanie prezentuje ostatni diagram postępu.

Przykład 5. Sumowanie liczb poprzez dekrementację i inkrementację
Mając pewne doświadczenie, możemy pokusić się o maszynę Turinga dodającą dwie liczby reprezentowane w systemie o dowolnej podstawie, przy czym szczegółowe działanie opiszemy dla liczb binarnych. Problem może być rozwiązany na wiele sposobów, a w naszym przypadku wykorzystamy wcześniej przedstawione maszyny Turinga. Skoro bowiem mamy maszyny Turinga zwiększające dane liczby o jeden (dla liczb binarnych przykład 2, dla liczb dziesiętnych przykład 3) i zmniejszające o jeden (maszyna z przykładu 4), wystarczy połączyć idee obu maszyn w jedną. Przyjmijmy nieograniczające założenie o rozdzieleniu dodajnej od dodajnika pojedynczym symbolem pustym, przyjmijmy ponadto że suma zapamiętana będzie w miejscu dodajnej a dodajnik zostanie wyczyszczony i dla liczb binarnych określmy maszynę Turinga o stanach $Q = \{ \mathsf{A}, \mathsf{B}, \mathsf{C}, \mathsf{D}, \mathsf{E}, \mathsf{F}, \mathsf{G}, \mathsf{Z} \}$, stanie początkowym $q_0 = \mathsf{A}$, stanie końcowym $F = \{ \mathsf{Z} \}$, alfabecie taśmy $\Gamma = \{ _, 0, 1 \}$ i symbolu pustym $b = _$. Tablicą sterującą będzie:

	A	B	C	D	E	F	G	Z
0	0 → A	0 → B	0 → C	1 ← D	_ ← E	0 ← F	1 ← A	
1	1 → A	1 → C	1 → C	0 ← F		1 ← F	0 ← G	
_	_ → B	_ ← E	_ ← D		_ ↓ Z	_ ← G	1 ← A	

zaś diagram stanów ma postać:



Rysunek 1.6 Diagram stanów maszyny Turinga sumującej dwie liczby binarne

Dla liczb w innych systemach alfabet taśmy obejmie musi być rozszerzony o niezbędne cyfry, ale zbiór stanów pozostanie dokładnie taki sam.

Zauważmy na wstępie opisu, że stan początkowy A służy ominięciu dodajnej i kiedy maszyna przejdzie w stan B głowica znajdzie się na pierwszym znaku dodajnika, co jest sytuacją wyjściową do maszyny dekrementującej z przykładu 4. Następnie zwróćmy uwagę na sterowanie wyróżnione czerwonym kolorem i zauważmy podobieństwo do maszyny Turinga z przykładu 4. Stany B, C, D powyższej maszyny wprost odpowiadają stanom A, B, C maszyny dekrementującej, natomiast stany końcowe wymagają osobnego omówienia. Maszyna dekrementująca w stanie odpowiadającym stanowi D omawianej maszyny odczytując symbol 1 kończyła pracę, natomiast w naszym przypadku przechodzimy do dalszych działań poprzez stan F. Natomiast stan niepoprawnego zakończenia w przypadku maszyny dekrementującej występujący w sytuacji samych znaków 0, w naszym przypadku skutkuje przejściem do stanu E czyszczącego

zera pozostałe po dodajniku i ostatecznie kończy działanie. Na zakończenie zauważmy, że sterowanie wyróżnione kolorem niebieskim odpowiada inkrementującej maszynie Turinga z przykładu 2, a różnica występuje w stanie końcowym. Maszyna inkrementująca kończyła pracę, natomiast rozpatrywana maszyna wraca do stanu A celem ponownego uruchomienia ominięcia dodajnej (teraz już sumy) z następującą dekrementacją.

Mimo oczywistej prostoty, opisana maszyna Turinga jest skrajnie nieefektywna. Na przykład dla dodania dwu liczb binarnych `1100101` oraz `11010011` (dziesiętnie 101 oraz 203) wykona aż 4494 kroki, co zresztą jest jednym z powodów rezygnacji z przedstawienia diagramów postępu.

Dla liczb o wyższych podstawach koniecznie będzie dodanie reguł omijających wszystkie możliwe cyfry w stanach A, B oraz C. Natomiast stan D wymaga dodania reguł odpowiednio zmniejszających cyfry o jeden oraz zastąpienia reguły zamieniającej `0` na `1` regułą zamieniającą `0` na najwyższą cyfrę systemu. Z kolei stany F oraz G wymagają dodania podobnych reguł jak odpowiadające im stany B oraz Z maszyny z przykładu 3.

Przykład 6. Pisemne dodawanie liczb binarnych

Istnieją bardziej efektywne algorytmy dodawania liczb niż z użyciem opisanej w poprzednim przykładzie dekrementacji/inkrementacji i można je zaimplementować w maszynie Turinga. Można na przykład spróbować zrealizować znany ze szkoły algorytm pisemnego dodawania, który dla przejrzystości przedstawimy w zastosowaniu do liczb binarnych.

Zaprezentowana maszyna Turinga będzie najtrudniejsza spośród zamieszczonych przykładów, dlatego przed jej określeniem i analizą wyjaśnimy ogólne działanie algorytmu. Przyjmujemy na wstępie, iż dane będą dwie liczby binarne oddzielone pojedynczym symbolem pustym, ponadto na początku działania głowica znajduje się na pierwszym niepustym symbolu z lewej strony, zaś efektem pracy będzie binarna reprezentacja sumy początkowych liczb w miejscu dodajnej z cyframi dodajnika zastąpionymi symbolami pustymi. Maszyna będzie dla kolejnych (począwszy od prawej strony) cyfr dodajnej zajmować się kolejnymi (również począwszy od prawej strony) cyframi dodajnika. Cyfry dodajnika będą zastępowane znakiem pustym, przy czym maszyna rozdzieli swoje działanie na przypadek gdy ostatnią cyfrą dodajnika jest znak `0` i na przypadek gdy ostatnią cyfrą dodajnika jest `1`. Rozdzielenie wydaje się naturalne z racji istotnie odrębnych sposobów działania, ponieważ w przypadku znaku `0` właściwa cyfra dodajnej nie ulegnie zmianie, natomiast w przeciwnym przypadku zmiana nastąpi.

Przy powyżej określonych ogólnych ramach algorytmu podstawowym problemem jest konieczność określenia położenia kolejnej cyfry dodajnej do ewentualnej zmiany. Dotychczasowe maszyny Turinga dokonywały zamian na końcach danych, a jeżeli zmiany sięgały do wnętrza, to ruch głowicy był jednokierunkowy, wymuszony wartością poprzedniego znaku i nie zachodziła potrzeba odszukania właściwego miejsca. Dlatego, oprócz koniecznych symboli `0`, `1` oraz symbolu pustego _ niezbędny będzie dodatkowy, roboczy symbol (przyjmijmy, że będzie to znak `a`) pozwalający wyróżnić i rozpoznać aktualną cyfrę do nadpisania. Mówiąc inaczej, maszyna Turinga zanim skasuje aktualnie ostatnią cyfrę dodajnika, musi wpierw w kolejne miejsce dodajnej wstawiać szczególny znak. Rozpocznie od najbardziej

prawej cyfry (rozpoznawanej po następującym po niej znaku pustym), by w dalszym przebiegu działań znak po lewej stronie szczególnego znaku uznać za znak do zastąpienia. Jednakże, zastępowanie cyfr dodajnej musi pamiętać cyfrę przed zastąpieniem, co w przypadku liczb binarnych implikuje dwa dodatkowe symbole robocze, odpowiednio dla cyfr `0` oraz `1`. Przyjmijmy zatem, że symbol `a` oznacza nadpisanie cyfry `0`, zaś symbol `b` oznacza nadpisanie znaku `1`.

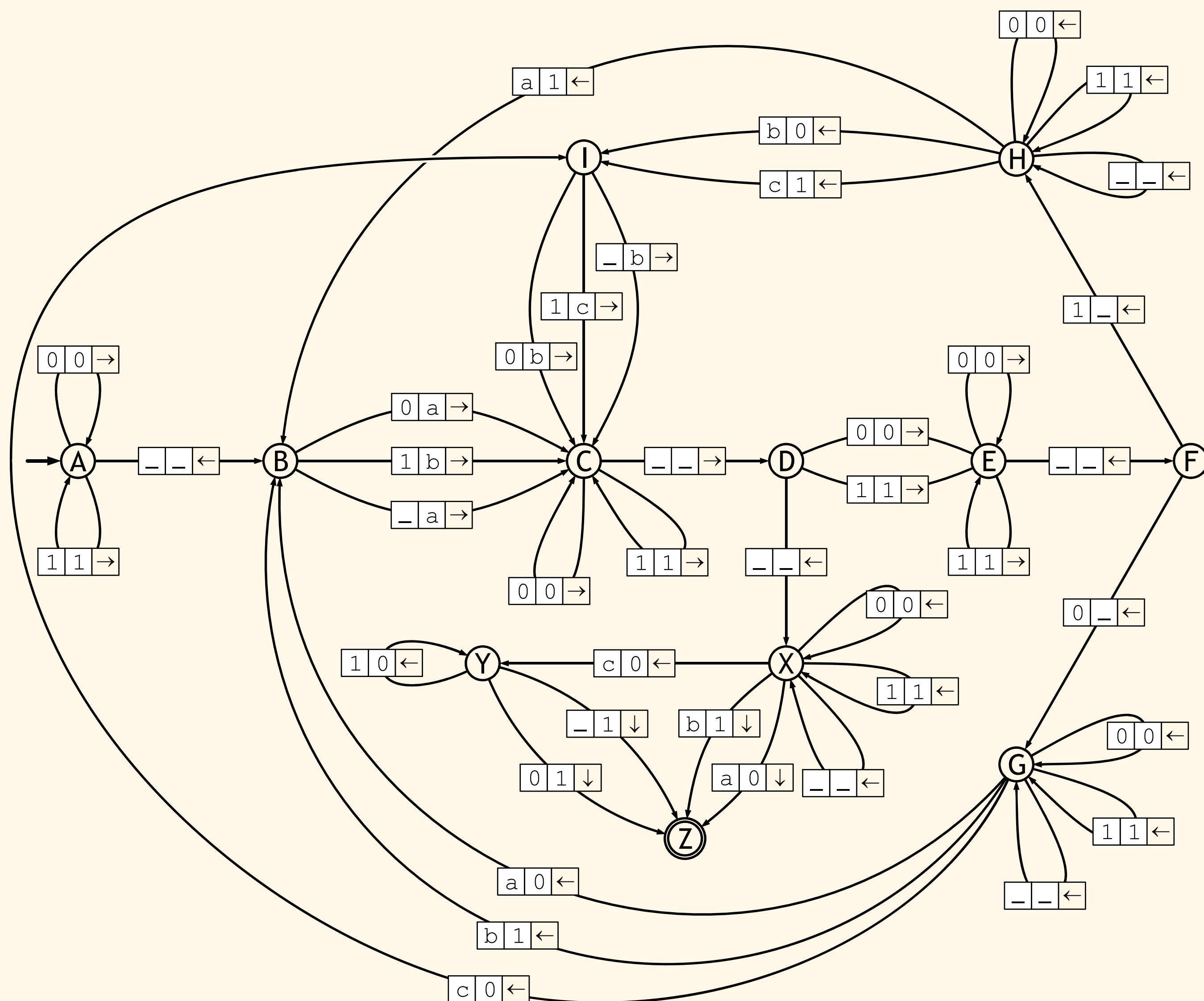
Nieco bardziej szczegółowy algorytm obsługi pojedynczego dodania dwu cyfr kształtuje się następująco: należy w miejscu kolejnej cyfry dodajnej (w postępie działań stającej się sumą) wpisać znak `a` lub znak `b` (odpowiednio dla odczytanej cyfry `0` lub cyfry `1`), następnie przejść na koniec aktualnej postaci dodajnika, skasować jego ostatnią cyfrę, wrócić do dodajnej (sumy) i w miejsce znaku `a` lub `b` wpisać znak `0` lub `1` w zależności od skasowanej cyfry dodajnika. Jak wspomnieliśmy wyżej, dwoistość końcowego znaku dodajnika wymusza dwa osobne działania. Dlatego jeżeli koniec dodajnika był cyfrą `0`, zaś w dodajnej widniał znak `a` należy wpisać `0`, jeżeli zaś w dodajnej widniał znak `b` należy wpisać znak `1`. Podobnie, jeżeli koniec dodajnika był cyfrą `1`, zaś w dodajnej widniał znak `a` należy wpisać `1`, podczas gdy jeżeli w dodajnej widniał znak `b` należy wpisać `0`. I tu pojawia się trudność. Wpisany znak `0` jest poprawny, ale merytorycznie pojawia się dodatkowa wartość, która musi być uwzględniona. Jeżeli kolejną cyfrą dodajnej będzie `0` problem możemy prosto rozwiązać wpisując znak `b` zamiast `a`, ale co zrobić gdy kolejną cyfrą dodajnej jest `1`? Wpisanie znaku `b` skutkowałoby utratą informacji o przeniesieniu, dlatego konieczny jest jeszcze jeden roboczy symbol i przyjmijmy, że będzie to znak `c`, oznaczające szczególne działanie.

Dodatkowy znak wymaga rozszerzenia sterowania w stosunku do opisanego powyżej, skutkującego dwoistą obsługą drukowania znaków pomocniczych. I tak jeżeli ostatnią cyfrą dodajnika było `0`, zaś znakami dodajnej było `a` lub `b` działanie jest zgodne z opisanym wyżej. Natomiast jeżeli znakiem dodajnej był znak `c`, drukujemy cyfrę `0` ale przechodzimy do szczególnej obsługi drukowania pomocniczych znaków, sprowadzającej się do wydrukowania znaku `b` w przypadku odczytu cyfry `0` i wydrukowania znaku `c` w przypadku odczytu cyfry `1`. Alternatywna obsługa gdy ostatnią cyfrą dodajnika było `1` tylko w przypadku odczytu symbolu `a` na dodajnej jest zgodna z wcześniej opisaną, to znaczy nie wymaga specjalnej obsługi drukowania dodatkowych znaków. W przypadku odczytu znaku `b` drukujemy znak `0`, ale przechodzimy do opisanej powyżej specjalnej obsługi drukowania. Podobnie w przypadku odczytu znaku `c` drukujemy znak `1` i również przechodzimy do specjalnej obsługi.

Podsumowując powyższe rozważania określmy maszynę Turinga o stanach $Q = \{$ A, B, C, D, E, F, G, H, I, X, Y, Z $\}$, stanie startowym $q_0 =$ A, stanie końcowym $F = \{$ Z $\}$, alfabecie taśmy $\Gamma = \{$ 0, 1, _, a, b, c $\}$, symbolu pustym $b =$ _ i tablicy sterującej:

	A	B	C	D	E	F	G	H	I	X	Y	Z
0	0 → A	a → C	0 → C	0 → E	0 → E	_ ← G	0 ← G	0 ← H	b → C	0 ← X	1 ↓ Z	
1	1 → A	b → C	1 → C	1 → E	1 → E	_ ← H	1 ← G	1 ← H	c → C	1 ← X	0 ← Y	
_	_ ← B	a → C	_ → D	_ ← X	_ ← F		_ ← G	_ ← H	b → C	_ ← X	1 ↓ Z	
a							0 ← B	1 ← B		0 ↓ Z		
b							1 ← B	0 ← I		1 ↓ Z		
c							0 ← H	1 ← I		0 ← Y		

Możemy również pokusić się o diagram stanów, tylko pozornie wydający się trudny w analizie:



Rysunek 1.7 Diagram stanów maszyny Turinga sumującej pisemnie liczby binarne

Spróbujmy skonfrontować implementację z wcześniejszymi rozważaniami opisując znaczenie każdego ze stanów. I tak poszczególne stany mają znaczenie następujące:

- A – służy jedynie początkowemu ominięciu cyfr dodajnej. Przechodząc w stan B maszyna znajdzie się na jej ostatniej cyfrze.
- B – początek zwykłej (niezwiązanej z przeniesieniem) obsługi pojedynczego dodania cyfr, polegające na nadpisaniu znaku pomocniczego.
- C – służy ominięciu już obsłużonych znaków dodajnej w dalszym przebiegu działań.
- D – pierwsza cyfra dodajnika, stan wyróżniony z racji konieczności uchwycenia końca działań.
- E – ominięcie cyfr dodajnika.
- F – skasowanie aktualnie ostatniej cyfry dodajnika warunkujące dalsze działanie. Stan G przy odczycie znaku `0` lub stan H przy odczycie znaku `1`.
- G – obsługa dodania cyfry `0` dodajnika. Ominięcie wszystkich znaków w lewo aż do napotkania znaków specjalnych. W przypadku odczytów znaków a lub b przejście do obsługi bez przeniesienia (stan B), zaś w przypadku znaku c przejście do obsługi z przeniesieniem (stan I)
- H – obsługa dodania cyfry `1` dodajnika. Ominięcie wszystkich znaków w lewo aż do napotkania znaków specjalnych. W przypadku odczytu znaku a przejście do obsługi bez przeniesienia, w przypadku odczytu stanów `b` lub `c` przejście do obsługi z przeniesieniem.
- I – szczególne nadrukowanie symboli specjalnych w związku z przeniesieniem.

Wyjaśnienia wymagają jeszcze dwa nie omawiane dotąd stany X oraz Y. Służą zakończeniu działań i obsłudze sytuacji dodajnej dłuższej od dodajnika. Jeżeli w stanie D cyfry dodajnika się skończą przechodzimy do stanu X który omija obsłużone cyfry sumy aż do napotkania szczególnych znaków `a` lub `b`, zastępowanych odpowiednią cyfrą. Natomiast w przypadku napotkania znaku `c` przeniesienie może objąć kolejne znaki `1`, skąd nadrukowania znaków `0` w stanie Y. W tym przypadku końcem działań jest pierwszy znak `0` lub koniec dodajnej.

Przedstawiona maszyna jest bardziej skomplikowana, ale dla dodania wcześniej wykorzystywanych liczb `1100101` oraz `11010011` (dziesiętnie 101 i 203) wykonane będzie tylko 197 kroków.

1.1.4 Związki z innymi dziedzinami

Przykład 7. Akceptacja słów języka regularnego

Maszyna Turinga potrafi w prosty sposób akceptować słowa języka regularnego, czyli języka dla którego istnieje gramatyka regularna generującego dany język. Prostota wynika z faktu równoważności istnienia gramatyki regularnej i istnienia automatu skończenie stanowego akceptującego dany język. Pojęcie automatu w ujęciu czysto teoretycznym nie jest przedmiotem naszych rozważań, dlatego poprzestaniemy na informacji iż przejście od gramatyki do automatu jest ściśle i formalnie określone i w ogólnych zarysach polega na zastąpieniu każdego symbolu nieterminalnego stanem, symbole terminalne stanowią symbole przejścia, zaś produkcje stanowią o przejściach miedzy stanami. W tym ograniczonym ujęciu maszyna Turinga jest równoważna automatowi skończonemu, zatem symbole nieterminalne gramatyki odpowiadają stanom maszyny Turinga, symbole terminalne odpowiadają alfabetowi wejściowemu, produkcje odpowiadają regułom sterowania. W sytuacji startowej na taśmie znajduje się spójny ciąg symboli alfabetu wejściowego.

Algorytm przekształcania gramatyki regularnej na maszynę Turinga (automat skończenie stanowy) mieści się w dziale informatyki zwanym Teoria języków formalnych i automatów i wykracza poza zakres przedmiotu, dlatego poprzestaniemy na prezentacji przykładu.

Rozpatrzmy zatem język:

$$L = \{ \mathtt{b}^n \mathtt{a} (\mathtt{cb})^k : n > 0, k \geqslant 0 \}$$

Dla tak określonego języka istnieje gramatyka regularna:

$$G = (N = \{ \mathsf{S}, \mathsf{A}, \mathsf{B}, \mathsf{C} \}, T = \{ \mathtt{a}, \mathtt{b}, \mathtt{c} \}, P, \mathsf{S})$$

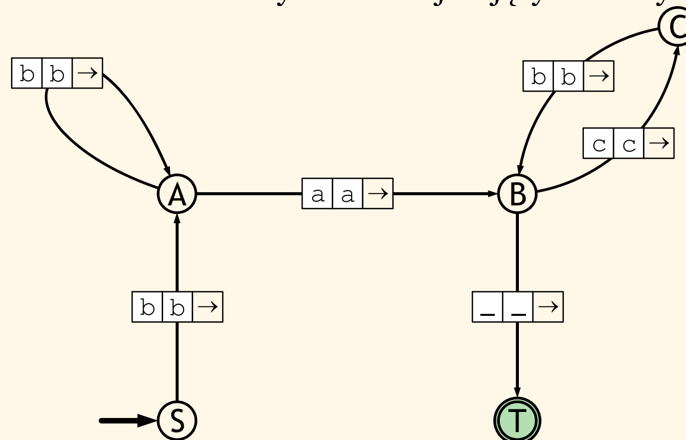
gdzie:

$$P = \left\{ \begin{matrix} \mathsf{S} \mapsto \mathtt{b}\mathsf{A}, & \mathsf{A} \mapsto \mathtt{b}\mathsf{A}, & \mathsf{A} \mapsto \mathtt{a}\mathsf{B}, \\ \mathsf{B} \mapsto \varepsilon, & \mathsf{B} \mapsto \mathtt{c}\mathsf{C}, & \mathsf{C} \mapsto \mathtt{b}\mathsf{B} \end{matrix} \right\}$$

Odpowiadająca powyższej gramatyce maszyna Turinga obejmuje zbiór stanów $Q = \{ \mathsf{S}, \mathsf{A}, \mathsf{B}, \mathsf{C}, \mathsf{T}, \mathsf{N} \}$. Stany S, A, B, C wprost odpowiadają symbolom nieterminalnym, natomiast stany T oraz N zostały dołożone tylko ze względów formalnych. Końcowy stan T odpowiada za akceptację danych wejściowych, a dla formalnej poprawności wprowadzamy końcowy stan N sygnalizujący brak akceptacji. Tym samym $q_0 = \mathsf{S}$, $F = \{ \mathsf{T}, \mathsf{N} \}$, alfabet taśmy obejmuje $\Gamma = \{ \mathtt{a}, \mathtt{b}, \mathtt{c}, _ \}$, przy czym pusty symbol $b = _$ jest również wprowadzony dla formalnej poprawności. Tablica sterująca ma postać:

	S	A	B	C
a	a ↓ N	b → B	a ↓ N	a ↓ N
b	b → A	b → A	b ↓ N	b → B
c	c ↓ N	c ↓ N	c → C	c ↓ N
_		_ ↓ N	_ ↓ T	_ ↓ N

przy czym zrezygnowaliśmy z kolumn stanów końcowych nie obejmujących żadnych reguł. Diagram stanów ...



Rysunek 1.8 Diagram stanów maszyny Turinga akceptującej język regularny

... przedstawia powyższy rysunek i w tym wypadku zrezygnowaliśmy z wyrysowaniu przejść do stanu N, ponieważ istotą działania maszyny jest akceptacja. Przyjmujemy, że we wszystkich sytuacjach nie zobrazowanych na diagramie występuje przejście do stanu N.

Jako przykład rozpatrzymy słowo `bbbacbcb`, przy czym dla prezentacji działania w zastępstwie diagramu postępu posłużymy się wywodami znanymi z działu poświęconego językom formalnym otrzymując:

$$S \xrightarrow{b} A \xrightarrow{b} A \xrightarrow{b} A \xrightarrow{a} B \xrightarrow{c} C \xrightarrow{b} B \xrightarrow{c} C \xrightarrow{b} B \xrightarrow{_} T$$

W uzupełnieniu jako przykład negatywny rozpatrzymy słowo `bab`, dla którego maszyna zakończy pracę w stanie N.

$$S \xrightarrow{b} A \xrightarrow{a} B \xrightarrow{b} N$$

Przykład 8. Rozpoznawanie palindromów

Gramatyki/języki regularne (a ściślej automaty skończenie stanowe) nie są jedyną strukturą znaną z teorii języków formalnych realizowaną poprzez maszynę Turinga. Przy spełnianiu dodatkowych warunków możliwe jest znalezienie maszyny Turinga równoważnej tak zwanemu deterministycznemu automatowi ze stosem, służącemu akceptacji gramatyk/języków bezkontekstowych. Niestety, brak prostego sposobu zamiany dowolnej gramatyki bezkontekstowej na automat ze stosem lub na maszynę Turinga, dlatego poprzestaniemy na przykładzie.

Rozpatrywanym językiem będzie język palindromów nad trzyznakowym alfabetem złożonym ze znaków a, b, c. Stosunkowo prosto można znaleźć odpowiednią gramatykę bezkontekstową generującą wyłącznie palindromy i będzie to gramatyka:

$$G = (N = \{ \mathsf{S} \}, T = \{ \mathtt{a}, \mathtt{b}, \mathtt{c} \}, P, \mathsf{S})$$

przy czym:

$$P = \left\{ \begin{array}{lll} \mathsf{S} \mapsto \mathtt{a}, & \mathsf{S} \mapsto \mathtt{a}\,\mathsf{S}\,\mathtt{a}, & \\ \mathsf{S} \mapsto \mathtt{b}, & \mathsf{S} \mapsto \mathtt{b}\,\mathsf{S}\,\mathtt{b}, & \mathsf{S} \mapsto \varepsilon \\ \mathsf{S} \mapsto \mathtt{c}, & \mathsf{S} \mapsto \mathtt{c}\,\mathsf{S}\,\mathtt{c}, & \end{array} \right\}$$

Łatwo zauważyć, że każda produkcja albo zawsze dokłada jednakowe symbole na końce, albo wstawia pojedynczy symbol do środka.

Maszyna Turinga realizująca zadanie akceptacji palindromów jest nieco bardziej skomplikowana i obejmuje zbiór stanów $Q = \{ \mathsf{P}, \mathsf{A}, \mathsf{B}, \mathsf{C}, \mathsf{D}, \mathsf{E}, \mathsf{F}, \mathsf{R}, \mathsf{T}, \mathsf{N} \}$, $q_0 = \mathsf{P}$, $F = \{ \mathsf{T}, \mathsf{N} \}$ (przy czym znaczenie stanów końcowych jest identyczne jak w dotychczasowych przykładach), $\Gamma = \{ \mathtt{a}, \mathtt{b}, \mathtt{c}, _ \}$, $b = _$ oraz tablica sterującą:

	P	A	B	C	D	E	F	R
a	_ → A	a → A	a → B	a → C	_ ← R	N	N	a ← R
b	_ → B	b → A	b → B	b → C	N	_ ← R	N	b ← R
c	_ → C	c → A	c → B	c → C	N	N	_ ← R	c ← R
_	T	_ ← D	_ ← E	_ ← F	T	T	T	_ → P

Zrezygnowano ze stanów końcowych, a ponadto dla skrócenia i przejrzystości pojedyncze litery N i T oznaczają przejścia do stanu opisanego literą bez zmiany symbolu taśmy i bez ruchu głowicy.

Pozornie złożone sterowanie realizuje nieskomplikowany algorytm polegający na kasowaniu identycznych znaków z końców ciągu symboli wejściowych z przejściem w stan akceptacji przy jego wyzerowaniu. Zarazem w przypad-

ku niemożności skasowania ostatniego symbolu identycznego ze skasowanym symbolem początkowym maszyna powinna przejść w stan błędu. Rodzaj działań określa pierwszy odczytany znak, stąd tak naprawdę maszyna posiada trzy niezależne zestawy stanów o analogicznym działaniu, wyróżnione kolorami. Dla przykładu poddajmy analizie sytuację gdy początkowym symbolem jest znak a. Maszyna wystartuje ze stanu P, skasuje znak a przechodząc do stanu A, w którym pozostanie przesuwając głowicę w prawo aż do końca wejściowego ciągu. Wyjdzie ze stanu A odczytując znak pusty z przejściem w stan D i z przesunięciem głowicy w lewo na ostatni znak wejściowy. Stan D skasuje znak, ale tylko gdy będzie to znak a i przejdzie do wspólnego stanu R powodującego powrót na początek ciągu z przejściem w stan P, a wiec do nowego cyklu kasowania. Jeżeli jednak w stanie D związanym z początkową literą a ostatnim znakiem jest inna litera maszyna zasygnalizuje błąd. Podobnie działa para stanów B oraz E ale dla znaku b, oraz para stanów C oraz F dla znaku c.

Jako przykład rozpatrzmy ciąg abbcbba, dla którego powrócimy do diagramów postępu, choć bez prezentacji kompletnego działania.

1.	P	_	**a**	b	b	c	b	b	a	_	_	_
2.	A	_	_	**b**	b	c	b	b	a	_	_	_
							...					
8.	A	_	_	b	b	c	b	b	a	**_**	_	_
9.	D	_	_	b	b	c	b	b	**a**	_	_	_
10.	R	_	_	b	b	c	b	**b**	_	_	_	_
							...					
15.	R	_	**_**	b	b	c	b	b	_	_	_	_
16.	P	_	_	**b**	b	c	b	b	_	_	_	_

27.	P	_	_	_	**b**	c	b	_	_	_	_	_
28.	B	_	_	_	_	**c**	b	_	_	_	_	_
29.	B	_	_	_	_	c	**b**	_	_	_	_	_
30.	B	_	_	_	_	c	b	**_**	_	_	_	_
31.	E	_	_	_	_	c	**b**	_	_	_	_	_
32.	R	_	_	_	_	**c**	_	_	_	_	_	_
33.	R	_	_	_	**_**	c	_	_	_	_	_	_
27.	P	_	_	_	_	**c**	_	_	_	_	_	_
34.	C	_	_	_	_	_	**_**	_	_	_	_	_
30.	F	_	_	_	_	_	_	**_**	_	_	_	_
	T	_	_	_	_	_	**_**	_	_	_	_	_

Początkowe działanie przedstawia lewy z powyższych diagramów, podczas gdy prawy diagram przedstawia sytuację końcową.

Spis rysunków

Indeks

A

D

L

M

P

S

T